28 | 沟通原则：什么时候应该妥协，什么时候应该坚持？

2020-07-20 臧萌

职场求生攻略 进入课程 >

讲述：臧萌
时长 13:00 大小 11.92M

你好，我是臧萌。本专栏进入到加餐模块啦，第一篇加餐，我们又来谈沟通了。

按照我们之前谈沟通的顺序，先是从工作的基本素质上，我们要重视沟通，主动沟通。然后从职场情商的角度，我们要重视沟通的方式方法，重视沟通的效果，尊重沟通的对象。从这两个方面来看的话，沟通还是正面与合作为主。但是我们在日常沟通里，面对的远不止这些和谐画面。

很多时候，我们还会遇到很多观念观点的冲突。如果冲突可以通过短时间的讨论达成一致，那就不能算有冲突。当然，也有可能大家心里的想法都不是很明晰，讨论下来也没一个明确的结果，这也不能算冲突。我们这里说的冲突，是双方或者多方的想法都非常明确，然后经过反反复复的讨论，大家的想法也无法达成一致的情况。

接下来在这一节里，我们就从处理冲突的角度来聊聊沟通。

有些冲突，程序员要主动退到一边

发生冲突的一个很大的原因，是有直接的利益冲突。比如谈地盘、谈系统归属、谈责任等等。这些情况是经理等管理者的主要战场，我们程序员打好辅助就可以，一般不需要主动露面谈。如果遇到这些相关的讨论找上自己，可以先简单应付，但是真正到了讨论核心问题的时候，一定要记得把自己的经理拉进来。

需要强调的一个点，千万不要做超出自己权利范围的承诺。大白话就是，自己说了不算的话，不要跟自己组之外的人说。

一般来说，大家都是会找对等的人来交流。比如说，如果这个会议上有对方的经理，那么自己组的经理也是要出面的。这不是排场面子的原因，而是对等的人，才能聊得下去，把事情敲定。举个简单的例子，一个国家的总统，和另一个国家的经济部长，是没办法聊的，因为很多事情经济部长无法拍板。

如何应对沟通中的冲突

属于我们程序员相关的冲突和争论，一般都是与技术和架构相关的。那么我们程序员应该如何应对沟通中冲突呢？我下面举几个例子，我们一块来看一下。

代码方面的个人偏好

每个人都有自己的偏好，比如说，有人喜欢用 Lambda，有人喜欢用 for 循环，有人喜欢这样去命名方法和变量，有人喜欢另外一种风格。从我个人的建议来说，谁负责写 code，谁就有权利决定用自己喜欢的风格写代码。只要功能可以完成，代码足够鲁棒，那外人就没权利多加干涉。毕竟我们不是流水线上的工人，不可能生产出一样的代码。

当然，不是说代码就没有标准。就 Java 来说，Java 编译器有 warning，还有 Sonar、PMD、Findbugs、Checkstyle 等工具。这些工具里有很多检查代码的规则，这些都是业内实际的标准和共识。从这个角度来说，你没必要用自己的个人偏好去指点别人。当然，如果被人指点，你也可以拿这个反击。毕竟行业内的标准都没说什么，你有什么资格说我？

接口的设计

接口的设计其实非常考验一个人的能力。它比具体的实现代码更抽象，但是比架构设计更具体，是连接两者的桥梁。

简单来说，接口就是程序世界的标准。标准的规定有一个公认的原则，就是要足够的包容。比如 Iterator、Collection 和 List 作为参数，兼容性逐渐降低。当然这个例子没什么大问题，大不了调用方自己转一下。但是如果用了 File 做参数，而实际上只需要 InputStream 读取数据，那调用方如果没有把数据存在 File 里，就不好弄了。

从参数的角度说开去，接口就是这样一个越包容，越优秀的东西。如何能用最少的约束，把事情描述清楚，就是接口的一大目的。面临接口设计的争论时，大家可以就这个点展开，各抒己见。最终得出结论，也不是难事儿。

系统架构

系统架构上的争论，则更是因人而异。因为做系统架构的过程，其实是每个人对一件事情理解和建模的过程。就好像扔给两个人一堆积木，让他们每人搭一个大楼，肯定俩人搭建的大楼不会一样。即使你指出方向，比如一个商业摩天大楼，外表也会不一样。即便是再规定得详细一点，比如要有广场、要有裙楼等等，细节部分也不会一样。那就更别说软件架构设计了，俩人架构的系统肯定不一样。

如果在软件架构上有冲突，应该怎么处理呢？软件架构结果的冲突只是表象。根源是大家对问题的理解，对系统如何应对问题的方式理解不同。这时候，如果只是纠结于架构的不同，是无法得出一个结论的。所以，为了解决冲突，应该大家都回退一步，先从问题入手，让大家对问题的理解能够统一，对解决问题的方式能够统一。这样，即使架构上有所不同，也是容易解决的。

当然，我在前面也提到过：令出多门，兵家大忌。如果就是不能统一呢？这种情况下，就得有一个总的架构师，能够维护整体解决方案的统一性，避免因为彼此的认识不同，导致系统无法统一。其实很多时候，系统架构很难说谁对谁错。但是如果各个子系统的风格不统一，那大概率就会有问题。

功能现在就要

很多时候，需求方会提出很多现在架构下不支持的功能。这时候我们就有两个选择，一个以快糙猛的方式，放弃遵循现在的架构，把功能怼出来。一个呢，是从长计议，深入理解

需求、升级架构，然后支持新的功能。

我来打个比方，系统架构就像是一张交通图。普通的需求就像是走高架，不需要绕路，没有红绿灯，可以用最短的时间到达。特殊的需求就像是走地面，甚至是走单行道，需要走走停停，甚至需要绕路，虽然费点劲，但是也能到，而且不和架构冲突。

架构不支持的需求，则是交通图上根本没有路的情况，就好比需要跨过一条河，但是河上并没有桥。这时候，第一种选择就是硬怼，比如放下车、找条船、甚至直接游泳过去。

第二种选择是先深入了解用户过河的需求，包括过河干什么、频次是多少、运载量是多少、能够接受的交通时间是多少等等。然后再根据需要来选择是修建码头、修建跨河大桥、还是修建隧道。

看似第一种选择是客户至上。但其实很多时候，第一种选择是短视的表现。一方面，你快糙猛做出来的东西，很可能无法完成用户接下来的需求。比如说，随着需求的发展，用户要求一天来回 1000 趟，每趟不能超过 3 分钟，游泳自然不可能。坐船的话，你去哪里找这么多的快船呢？换句话说，违背现有架构，快糙猛地实现客户的需求，是对客户的溺爱，而非真的的客户至上。

另一方面，没有深入理解用户的需求，其实会错过和客户一起深入理解系统和业务的机会。系统架构不支持，确实是系统本身的限制。但是如何让系统更完美呢？就是你要通过这样的机会，深入思索业务模型的限制，通过升级系统来让系统建模更加通用，可以应对客户以后的新需求。

方式方法可以妥协，原则和目标必须坚持

上面的几个例子中，一个共性是大家先后退寻找共识（common ground），然后以此为基础，把事情向前推进。这里我总结一下上面几个例子里所说的共识。

1. 代码偏好：findbugs 等业内标准。
2. 接口设计：接口设计应该包容。
3. 系统架构：大家对问题的理解要统一。
4. 新功能：理解新需求，升级架构，以支持新功能。

回到我们标题上提出的问题，我们在交流中遇到冲突时，什么时候应该妥协，什么时候应该坚持呢？这里我给出一个自己的标准，那就是做事情的方式方法可以妥协，但是目标本身必须坚持。

对比上面的四个例子你就能发现，所谓的标准，越来越模糊。什么是包容、什么是对问题的理解、什么又是新功能的本质，越来越没有绝对的正确。每个人，在自己不同的阶段，都会有不同的理解，更不用说在不同的人之间。

这就带来一个实际的问题，就是冲突的情况不可避免。这时候我们程序员千万不要怕冲突，只要你是冲着做事情去，而不是冲着搞事情去，就一定要将自己心中的疑惑说出来。

说出来的好处，首先是可以互相切磋，提升彼此的认识。正所谓不打不相识，很多时候大家的认识都不够全面，通过冲突的方式，可以让彼此都拓展自己的认识。

其次，要说出来可以逼迫自己深入思考。想和别人刚，自己得先把自己的想法理清楚。很多时候，自己的想法是想当然的，当这种想当然和别人发生冲突的时候，就需要自己深入思考，或者找出自己的理由，或者放弃自己的成见，接受别人的观点。无论是哪种，对自己来说都是成长。

最后，有疑惑就说出来，可以让彼此合作更默契。我们经常说一个团队有默契，其实默契就是这样磨合出来的。不磨合，只是表面的迎合，并不能让一个团队里的成员配合无间。

总结

我想特别强调的一点是，程序员不要一味地逃避冲突。你如果用妥协来避免冲突，这绝对不能很好地解决问题。当然，我不是说要做一个“杠精”，而是要在做事情的基础上，充分表达自己的观点和认识。

如果自己明知道别人的做法有问题，但是你选择不表达自己的看法，长久下来，只会让别人觉得你没想法。和自己有关的东西，更要在自己不认同的时候发声。如果你只是一味地委屈自己，成全别人，其实只会让人觉得你软弱。

很多程序员不喜欢交流，不喜欢拒绝别人，更不要说和人争论、冲突。说实话，我也非常不喜欢和人冲突。但是正是自己这种避免冲突的倾向，让我吃了不少亏。举个例子，我之

前觉得架构设计有问题，我也不会说什么，觉得只要能做出来，实现功能就可以。后来发现，这些架构设计的问题，会慢慢变成坑，变成阻碍一个组发展的绊脚石。

现在我只要在工作中遇到觉得有问题的事情，马上就会要求展开细聊。如果其他人时间不够，就另外找时间聊。我从原来逃避冲突的人，变成了主动制造冲突的人。

根据我的观察，这种冲突，往往都能够让事情得到更好的结果。如果一味地为了避免冲突而保持缄默，其实长期来看，是无法融入到一个团体的。当然，如果一个团体里经常就相同的事情冲突来冲突去，那很可能说明大家就是无法磨合得很好。在这种情况下，如果你不打算离开这个团体，在面对冲突的时候千万别打退堂鼓。

极客时间

思考题

你在工作中，有遇到过哪些让你记忆深刻的冲突？这些冲突是以双方都满意结束，还是以双方不欢而散结束呢？你有哪些处理冲突的经验和教训？

好了，今天的课到这里就结束了。欢迎你在评论区和我交流。也欢迎把这篇文章分享给你的朋友或者同事，一起交流一下。

提建议

更多课程推荐

© 版权归极客邦科技所有，未经许可不得传播售卖。 页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 27 | 答疑篇：什么样的技术观能够更快成长？

下一篇 29 | 加班：加班逃不过，如何用正确姿势加班？

精选留言 (3)

写留言

Geek_3b1096
2020-07-20

紧张语速就快导致本来想好的东西说得乱七八糟

展开

作者回复: 哈哈哈哈哈哈，你这个留言带音效。

顺着你的思路说一下，重要的交流之前，不妨理一理自己的思路，列出要说的要点。不要被别人打乱节奏，导致自己想说的没说出来。

2

Sdylan
2020-07-20

让我想起一件冲突的事，18年的时候做了一个公共模块，接入方是上海的一个团队。需要个性化的东西。他们说要这么做，我说那么做是存在问题的，第三方平台不支持这么玩。后面对方的经理出来，我也是这么说，她后面来了一句，我和你们部门长很熟，你找你们xx经理来。那会入职不久，当面就刚了，后来有事没有继续电话沟通。隔一天之后，就直接被对方那个经理邮件投诉，不配合。后面经理出马、我负责说技术，他负责圆。...

展开

作者回复:
三点总结，个个经典。

从这个事情上，也能看出沟通的重要性。能把自己心里想的事情，整理清晰，表述清晰，让别人正确的理解，是一个非常非常重要的能力。

心态问题也很重要。关键的一点事，有理不在声高。心平气和的说话，😄。

2

bigben
2020-07-22

没有加班福利就不加班

展开

作者回复:

有也不一定值得加班，自己能选择最好。